Modifying K Nearest Neighbor for Content based Word Classification by Graph Similarity Metric

Taeho Jo
President
Alpha AI Publication
Cheongju, South Korea
tjo018@naver.com

***Abstract*—This article proposes the modified KNN (K Nearest Neighbor) algorithm which receives a graph as its input data and is applied to the word categorization. The graph is more graphical for representing a word and the synergy effect between the text categorization and the word categorization is expected by combining them with each other. In this research, we propose the similarity metric between two graphs representing words, modify the KNN algorithm by replacing the exiting similarity metric by the proposed one, and apply it to the word categorization. The proposed KNN is empirically validated as the better approach in categorizing words in news articles and opinions. In this article, a word is encoded into a weighted and undirected graph and it is represented into a list of edges.**

I. Introduction

Word categorization refers to the process of classifying each word into a particular category or some categories among the predefined ones as an instance of classification task. As the preliminary tasks, we must predefine a finite number of categories and prepared words which are labeled one of the predefined ones as sample words. The labeled sample words are encoded into their structured forms and the classification capacity is constructed by learning them. Novice words are encoded into their structured forms and classified into one or some of the predefined categories. In this research, we assume that the supervised learning algorithms are used as the approach to the word categorization.

We mention the facts which motivates for doing this research. Encoding words into numerical vectors with many features for the robustness causes much computation time by the huge dimensionality [4]. The sparse distribution in each numerical vector which represents a text or word degrades the discriminations among numerical vectors [4]. Representing knowledge into graphs which are called ontology or word net as the structured forms which are understandable by computers became very popular trend [2][31]. Therefore, in this research we attempt to represent words into graphs, motivated by the facts.

Let us mention some ideas which are proposed in this research, motivated by the above facts. In this research, each word is encoded into a graph where its vertices indicate text identifiers and its edges indicate their relations. We define the similarity measure between two graphs which is specialized for doing the word categorization task. Using the similarity measure, we modify the KNN (K Nearest Neighbor) into its graph based version where a graph is given as the input data, and apply it as the approach to the word categorization. Hence, this research provides the solution to the above problems in encoding words into numerical vectors, and corresponds to the popular knowledge representations.

Let us mention some benefits which are expected from this research. We expect more compact representations of words by avoiding the huge dimensionality from encoding them into numerical vectors. We expect more discrimination among representations of words by avoiding completely the sparse distribution among numerical vectors representing words. We may expect the better word categorization performance by solving the problems in encoding words into numerical vectors. However, we need to define more operations on graphs, referring to the developed manipulations on the ontologies and the graph theory.

Let us mention the organization of this research. In Section II, we explore the previous works which are relevant to this research. In Section III, we describe in detail what we propose in this research. In Section IV, we validate empirically the proposed approach by comparing it with the traditional one. In Section V, we mention the significance of this research and the remaining tasks as the conclusion.

II. Previous Works

This section is concerned with the previous works which are relevant to this research. In Section II-A, we explore the previous cases of applying the KNN algorithm to text mining tasks. In Section II-B, we survey the schemes of encoding texts or words into structured data. In Section II-C, we describe the previous machine learning algorithms which receive alternative structured data such as tables and string vectors to numerical vectors. Therefore, in this section, we provide the history about this research, by surveying the relevant previous works.

A. Applications to Word Classification and its Derived Tasks

This section is concerned with the previous works on applying the modern versions of KNN algorithm to the word classification tasks. The KNN algorithm is modified into versions where other types of structured data are used as the input data. The modified KNN algorithms are applied to the word classification tasks: topic based word categorization, keyword extraction, and index optimization. We present the trials of modifying the KNN algorithm into the versions which solve the problems in encoding words into numerical vectors. This section is intended to explore the previous works on modifying and applying the KNN algorithm for the word classification tasks.

Let us explore the previous works which used the modernized KNN algorithm for the topic based word categorization. The cosine similarity was modified into the similarity metric which considers the similarity among features in modernizing the KNN algorithm [9]. It was proposed that words should be encoded into tables, instead of numerical vectors, in using the KNN algorithm [11]. The KNN algorithm was modified into the version which receives a string vector, instead of a numerical vector, in using it for the topic based word categorization [12]. In the above literatures, the modernized KNN algorithms were used for the topic based word categorization with the empirical validations of their better performance.

The keyword extraction may be considered as an instance of word categorization. The KNN algorithm where the similarity metric which considers the similarities among features was used, was applied to the keyword extraction [13]. Another modernized KNN algorithm which classified a table directly, was used for the keyword extraction [14]. One more modernized KNN algorithm which processes string vectors directly was mentioned as the approach to the keyword extraction [15]. In the above literatures, the keyword extraction was mapped into the binary classification where each word is classified into keyword or non-keyword.

Let us mention the previous works on applying the modernized KNN algorithms to the index optimization which is derived from the word categorization. The KNN version which considers the similarities among features was applied to the index optimization in [16]. The KNN version which classifies a table directly, instead of numerical vectors, was used to the index optimization in [17]. The version which receives a string vector as its input data was proposed as the approach to the index optimization [18]. The index optimization was interpreted into the instance of word classification in the above previous works.

Let us mention some distinguished points of this research from the above ones. We surveyed the works where the modernized versions of KNN algorithm were applied to the word categorization and its related tasks. We mentioned the three kinds of modernized versions of KNN algorithm; the improved numerical vector based version, the table based version, and the string vector based version. The proposed version of KNN algorithm is one which classifies a graph which represent a word directly. It will be applied to the word categorization for validating its performance, empirically.

B. Word and Text Encoding

This section is concerned with the previous cases of encoding words and texts into non-numerical vectors. The problems in encoding texts or words into numerical vectors were discovered in previous works. There were trials to solve the problems by encoding them into other types of structured data. We mention the tables, the string vectors, and the graphs as representations of words or texts which are alternative to the numerical vectors. This section is intended to survey the previous cases of encoding texts or words into non-numerical vectors in other tasks.

Let us mention the previous works on encoding texts or words into tables. Words were encoded into tables, in using the AHC algorithm for clustering words [19]. Texts were encoded into tables in applying the KNN algorithm to the text categorization [20]. Texts were encoded so in using the AHC algorithm for clustering texts [23]. In the above literatures, texts or words were encoded into tables in using the KNN algorithm and the AHC algorithm.

Let us consider the previous cases of encoding texts or words into string vector. It was proposed that words should be encoded into string vectors for applying the AHC algorithm for clustering words [21]. It was proposed that texts should be encoded into string vectors for applying the KNN algorithm for categorizing texts [22]. Encoding texts so was proposed for applying the AHC algorithm for clustering texts [24]. The cases of encoding texts or words into string vectors are presented in the above literatures.

Let us consider encoding words or texts into graphs by the influence of social mining [3]. Words were encoded into graphs in applying the AHC algorithm to the word clustering [10]. Texts were encoded into graphs in applying the KNN algorithm to the text categorization [25]. In applying the AHC algorithm to the text clustering, texts were encoded so [26]. In the above literatures, we presented the previous cases of mapping raw data into graphs.

We mentioned the three schemes of encoding words or texts into structured in the previous works. We adopt the third scheme where words are encoded into graphs, in this research. We define the similarity metric between graphs, and modify the KNN algorithm into the version which processes graphs directly. We apply the modernized KNN algorithm for implementing the word categorization system. In this study, we validate empirically the modernized version by comparing it with the traditional version in the word categorization.

C. Non-Numerical Vector based Machine Learning Algorithms

This section is concerned with the previous works on non-numerical vector based machine learning algorithms. In the previous section, we explore the cases of encoding words or texts in using the KNN algorithm or the AHC algorithm. In this section, we mention the string kernel based Support Vector Machine, the table matching algorithm, and the Neural Text Categorization, as non-numerical vector based machine learning algorithms. In the traditional machine learning algorithms, it is assumed that input data is absolutely given as a numerical vector, whereas in what mentioned in this section, input data is given as alternative structured data. This section is intended to explore the works which are involved in proposing the kind of machine learning algorithms as the approaches to the text categorization.

In previous works, the string kernel based SVM (Support Vector Machine) was proposed as a classifier. It was initially proposed as the approach to the text classification by Lodhi et al. in 2002 [30]. It was used for classifying proteins in the bioinformatics by Leslie et al. in 2006 [29]. It was applied to the sentence classification by Kate and Mooney in 2006 [28]. Because it costs too much time for computing the lexical similarity between texts, the string kernel based SVM is infeasible to classify long texts.

Let us mention the table based matching algorithm as a non-numerical vector based classification algorithm. It was initially proposed as the approach to the text categorization by Jo and Cho, in 2008 [27]. It was applied to the soft categorization of texts by Jo in 2008 [5]. It was improved into the more stable approach to the text categorization by Jo in 2015 [8]. In using the table based matching algorithm for categorizing texts, texts are encoded into tables as non-numerical vectors.

Let us consider the Neural Text Categorizer specialized for the text categorization; in it, texts are encoded into string vectors. It was initially proposed as the approach to the text categorization by Jo in 2008 [6]. It was applied to both soft and hard categorization of texts in 2010 [7]. It was used for classifying texts in Arabian by Abainia et al. in 2015 [1]. It was mentioned as an innovative neural network model by Vega and Mendez-Vasquez [32].

Let us mention the three classification algorithms as non-numerical vector based approaches to the text classification. We mentioned raw texts, tables, and string vectors as structured forms which are processed by the above classification algorithm. In this research, words are encoded into graphs and the similarity between two graphs is defined. The KNN algorithm will be modified into the version which processes graphs directly. We empirically validate the modified KNN algorithm by comparing it with the traditional one in the word categorization.

III. Proposed Approach

This section is concerned with encoding words into graphs, modifying the KNN (K Nearest Neighbor) into the graph based version and applying it to the word categorization, and consists of the three sections. In Section III-A, we deal with the process of encoding words into graphs. In Section III-B, we describe formally the process of computing the similarity between to graphs. In Section III-C, we do the graph vector based KNN version as the approach to the word categorization, and In Section III-D, we present the word classification system where the proposed KNN version is adopted. Therefore, this section is intended to describe the proposed KNN version as the word categorization tool.

A. Word Encoding

This section is concerned with the process of transforming words into graphs. A graph is defined as the two sets: vertex set and edge set. A word is encoded into a graph with the three steps: vertex set definition, edge set definition, and edge weighting. In the graph which represents a word, vertices are text identifiers which include it, and edges are similarities among texts. This section is intended to describe the three steps which are presented in Figure 1-3 and are involved in encoding words.

The process of defining vertices as the first step of encoding words into graphs is illustrated in Figure 1. It is assumed that a word is given as an input and a corpus is initially prepared. Texts which include the word are extracted from the corpus. A list of texts becomes a set of vertices for construction the graph. If too many text are extracted from the corpus, we may select texts where words are weighted highly than the threshold.

The process of defining a set of edges for representing a word into a graph is illustrated in Figure 2. In the previous step, texts which are related with the word are extracted as a set of vertex. The complete graph is defined among texts which are given as vertices and some edges with lower weights are eliminated. Each edge between two texts is regarded as a similarity or a distance between them. If very small number of texts is set as vertices, complete links may be considered.

The process of computing an edge weight between texts is illustrated in Figure 3. N texts which are relevant to the word are defined as vertices, and similarities among texts are defined as edge weights. The complete links among N texts are given as a $N \times N$ matrix whose diagonal elements are given as 1.0s and whose off-diagonal elements are given as similarities which are computed by equation in Figure 3. We consider off-diagonal elements as weights of edge candidates and select edges with higher weights than the threshold among them. The $N \times N$ matrix which is presented in Figure 3 is the representation of complete links among vertices.

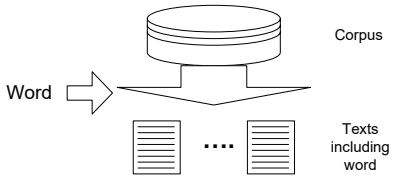

Figure 1. Word Indexing

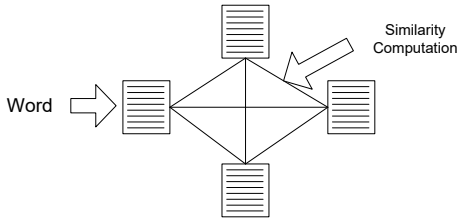

Figure 2. Word Representation: Graph

A word is encoded into a graph with the three steps which are presented in Figure 1-3. Texts which are related with the word are given as vertices, and similarities among texts are given as edges, in the graph which represents a word. The graph is viewed as a set of edges each of which consists of two vertices and their similarity, in the implementation level. It is assumed that the weight in each edge which indicates a similarity between texts is always given as a normalized value between zero and one. We need to define the operations on graphs for modifying machine learning algorithms into the versions which process them directly.

B. Graphs Similarity

This section is concerned with the similarity metric between two graphs. In the previous section, we mentioned the process of encoding words into graphs. We need to define the similarity metric between graphs, for modifying the KNN algorithm into the version which processes graphs directly. In this study, we view a graph as an edge set and define similarities among edges, in order to do that. This section is intended to describe the process of computing the similarity between graphs.

Let us mention the computation of similarity between two edges as the basis for computing one between two graphs. Each edge is expressed as an entry of three values as shown in equation (1),

$$e \equiv (node_1, node_2, weight) \tag{1}$$

the two edges, e_1 and e_2 are expressed as equation (2) and (3),

$$e_1 = (node_{11}, node_{12}, weight_1) \tag{2}$$

$$e_2 = (node_{21}, node_{22}, weight_2) \tag{3}$$

and $weight_1$ and $weight_2$ are given as normalized values between zero and one.

We consider the three possible cases between two edges as both nodes are same to each other, either of them is same, and neither of them is so. The similarity between two edges is defined on the three conditions:

- In the two edges, if both nodes are same to each other, the similarity between them is defined by equation(4),

$$sim(e_1, e_2) = \frac{1}{2}(weight_1 + weight_2) \tag{4}$$

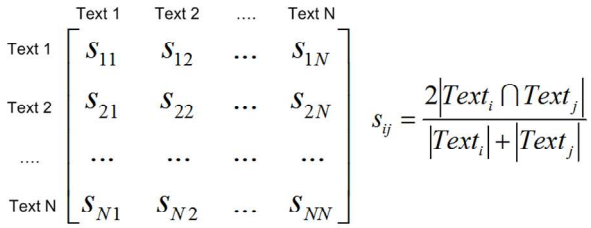

Figure 3. Similarity Matrix

- In the two edges, if only either of two nodes are same to each other, the similarity between them is defined by equation(5),

$$sim(e_1, e_2) = (weight_1 \times weight_2) \tag{5}$$

- In the two edges, if no node are same to each other, the similarity between them is zero.

The edge similarity will be used for computing the similarity between an edge and a graph, next.

The similarity between two edges is expanded into one between an edge and a graph. The graph, G is expressed as a set of edges, $G = \{e_1, e_2, \ldots, e_{|G|}\}$. The similarity, $sim(e_i, G)$, is computed by equation (6),

$$sim(e_i, G) = \max_{k=1}^{G} sim(e_i, e_k) \tag{6}$$

The similarity between an edge and a graph is the maximum among its similarities with ones in the graph, as shown in equation (6). When only edge e_r in the graph, G, have both identical, assuming that all weights are constant between zero and one, the similarity between an edge and a graph is expressed by equation (7),

$$sim(e_i, G) = sim(e_i, e_r) \tag{7}$$

The similarity between an edge and a graph is expanded into one between two graphs,further. The two graphs are notated by G_1 and G_2, and they are viewed as edge sets, as shown in equation (8) and (9),

$$G_1 = \{e_{11}, e_{12}, \ldots, e_{1|G_1|}\} \tag{8}$$

$$G_2 = \{e_{21}, e_{22}, \ldots, e_{2|G|_2}\} \tag{9}$$

For each edge in the graph, G_1, its similarity with the graph, G_2 is computed by equation (6). The similarity between the two graphs, G_1 and G_2 is computed by equation (10),

$$sim(G_1, G_2) = \frac{1}{|G_1|} \sum_{i=1}^{|G_1|} sim(e_{1i}, G_2) \tag{10}$$

The similarity between the two graphs, G_1 and G_2, is always given as a normalized value between zero and one.

We mentioned the similarity between two graphs as a normalized value between zero and one, and let us assume that all edges are weighted as 1.0 in both graphs. If $G_1 = G_2$, the similarity between two graphs is given as 1.0 by equation (11),

$$\begin{aligned} sim(e_{1i}, G_2) &= 1.0 \\ sim(G_1, G_2) = \frac{1}{|G_1|} \sum_{i=1}^{G_1} sim(e_{1i}, G_2) &= \frac{|G_1|}{|G_1|} = 1.0 \end{aligned} \tag{11}$$

If no vertex shared by two graphs, the similarity between two graphs given as zero by equation (12),

$$\begin{aligned} sim(e_{1i}, G_2) &= 0.0 \\ sim(G_1, G_2) = \frac{1}{|G_1|} \sum_{i=1}^{G_1} sim(e_{1i}, G_2) &= \frac{0}{|G_1|} = 0.0 \end{aligned} \tag{12}$$

The similarity between two graphs is always given as a normalized value between zero and one by equation (13),

$$\begin{aligned} G_1 \cap G_2 \subseteq G_1, G_1 \cap G_2 \subseteq G_2 \\ 0 \leq sim(e_{1i}, G_2) \leq 1.0 \\ 0 \leq \frac{1}{|G_1|} \sum_{i=1}^{G_1} sim(e_{1i}, G_2) \leq 1.0 \\ 0 \leq sim(G_1, G_2) \leq 1.0 \end{aligned} \tag{13}$$

Each edge is usually weighted between zero and one, so the similarity between two graphs is clearly given as a normalized value.

C. Proposed Version of KNN

This section is concerned with the graph based version of the KNN algorithm which is illustrated in Figure 4. We described the process of encoding words into graphs in Section III-A, and assume that training examples and a novice example are given as graphs. We apply the similarity

metric between two graphs which is described in Section III-B, to selection of nearest neighbors. The novice item is classified by voting ones of nearest neighbors, and variants may be derived by manipulating the selection scheme and the voting one. This section is intended to describe the proposed version of the KNN algorithm which processes graphs directly and its variants.

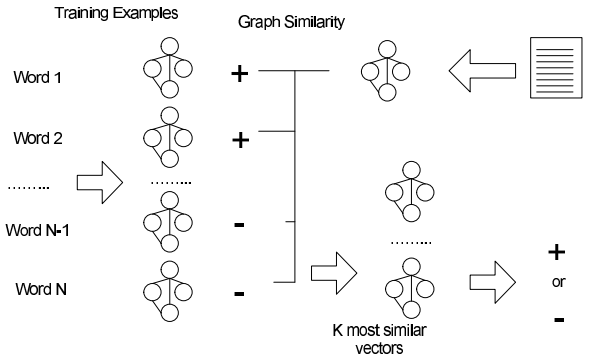

Figure 4. Proposed Version of KNN Algorithm

Let us mention the process of selecting nearest neighbors as the references for classifying a novice item. The sample words and a novice word are mapped into graphs by the process which was covered in Section III-A. The similarities of the novice item with the training ones are computed by the equation which is defined in Section III-B. The training examples are ranked by their similarities and the K most similar ones are selected as nearest neighbors. We adopt the rank based scheme of selecting the nearest neighbors in the KNN algorithm.

Let us mention the process of voting the labels of the nearest neighbors for deciding one of a novice item. We notate the set of nearest neighbors of the novice item, G , whose elements are given as tables and their target labels, by equation (14),

$$Ne_k(G) = \{(G_1, y_1), (G_2, y_2), \ldots, (G_k, y_k)\}, \quad y_i \in \{c_1, c_2, \ldots, c_m\} \tag{14}$$

where $c_1, c_2, \ldots, c_m$ are the predefined categories and k is the number of nearest neighbors. The number of the nearest neighbors which are labeled with the category,c_i is notated by $Count(Ne_k(G), c_i)$. The label of the novice item, G, is decided by the majority of categories in the nearest neighbors, as expressed by equation (15),

$$c_{\max} = \underset{i=1}{\overset{m}{\operatorname{argmax}}} Count(Ne_k(G), c_i) \tag{15}$$

The external parameter,k, is usually set as an odd number for avoiding the possibility of largest number of nearest neighbors to more than one category.

Let us mention the weighted voting of labels of nearest neighbors as the alternative scheme to the above. Assuming that the similarity between two tables as a normalized value between zero and one, and we may use the similarities with the nearest neighbors, $sim(G, G_1), sim(G, G_2), \ldots, sim(G, G_k)$ as weights, $w_1, w_2, \ldots, w_k$ by equation (16),

$$w_i = sim(G, G_i) \tag{16}$$

indicates the similarity of a novice table with the ith nearest neighbor. The total weight of nearest neighbors which labeled with the category, c_i by equation (17),

$$Weight(Ne_k(G), c_i) = \sum_{G_j \in c_i}^{k} w_j \tag{17}$$

The label of the novice item, G, is decided by the category which corresponds to the maximum sum of weights as shown in equation (18),

$$c_{\max} = \underset{i=1}{\overset{m}{\operatorname{argmax}}} Weight(Ne_k(G), c_i) \tag{18}$$

When the weights of nearest neighbors are set constantly, equation (18) is same to equation (15), as expressed in equation (19),

$$Weight(Ne_k(G), c_i) = Count(Ne_k(G), c_i) \tag{19}$$

We described the proposed version of the KNN algorithm in this section. In using the proposed KNN algorithm, raw data is encoded into graphs, instead of numerical vectors. The similarities of a novice item with the training examples are computed by the similarity metric which is defined in Section III-B. The rank based selection is adopted as the scheme of selecting nearest neighbors among training examples. Because we are interested in the comparison of the traditional version and the proposed version as the ultimate goal, we use the unweighted voting in the experiments which are covered in Section IV.

D. Word Classification System

This section is concerned with the word classification system which adopts the graph based KNN algorithm. We described the proposed version of KNN algorithm as the approach for implementing the system, in Section III-C. The preliminary tasks are to predefine categories as a list and to gather sample labeled words. The main functions of the system are to encode words into graphs and to classify them into one or some among predefined categories. This section is intended to describe the word classification system with respect to its architecture and its function.

The sample words are illustrated for implementing the topic based word classification by the proposed KNN algorithm in Figure 5. The topics are predefined as topic 1, topic 2, ..., topic M. The N words are allocated for each topic as the sample words. The balanced distribution over the categories is necessary for preventing the bias toward

a particular topic. $M \times N$ sample words are encoded into tables by the process which is mentioned in Section III-A.

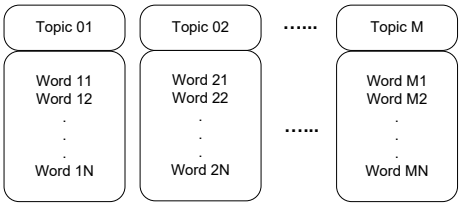

Figure 5. Sample Words

The entire architecture of the proposed word categorization system is illustrated in Figure 6. The sample words which are labeled with one of M categories and the unlabeled ones as novice items are encoded into graphs. For each novice graph, its similarities with the sample graphs are computed by the metric which is mentioned in Section III-B, in the similarity computation module, and the k most similar sample ones are selected as the nearest neighbors. The label of the novice item is decided by voting ones of nearest neighbors in the voting module. This system consists of the three components: the encoding module, the similarity computation module, and the voting module.

The execution process of the proposed system is illustrated in Figure 7. The sample words which are collected by the process mentioned above and the word which is given as the input are encoded into graphs. Its nearest neighbors are extracted from the samples through the similarity computation module. The category of the novice word is decided by voting ones of the nearest neighbors. The category of the novice word is decided as the final output in the system.

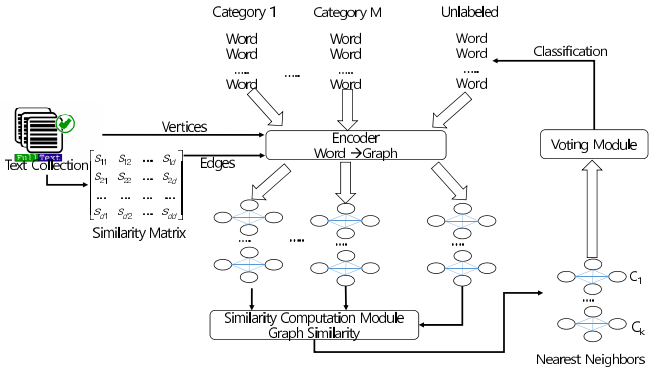

Figure 6. Proposed System Architecture

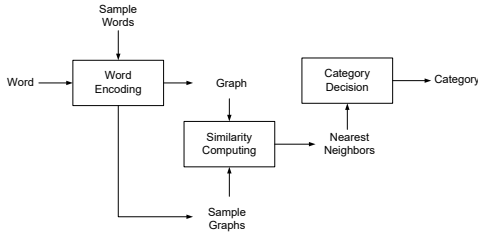

Figure 7. Execution Process of Proposed System

Let us make some remarks on the proposed system which is illustrated in Figure 6 as its architecture. Words are encoded into graphs, instead of numerical vectors. Graphs which represent novice words are classified directly by the proposed KNN algorithm. The classification performance is improved by what proposed in this research, as shown in Section IV. In the next research, we present the graphical user interface and the source code which are necessary for implementing the system as a complete one.

IV. Experiments

This section is concerned with the empirical experiments for validating the proposed version of KNN, and consists of the five sections. In Section IV-A, we present the results from applying the proposed version of KNN to the word catego-

rization on the collection, NewsPage.com. In Section IV-B, we show the results from applying it for categorizing words from the collection, Opinosis. In Section IV-C, we mention the results from comparing the two versions of KNN with each other in categorizing words from 20NewsGroups.

A. NewsPage.com

This section is concerned with the experiments for validating the better performance of the proposed version on the collection: NewsPage.com. The four categories are predefined in this collection and from the collection, NewsPage.com, we gathered the words category by category as the labeled ones. Each word is allowed to be classified into only one of the four categories. In this set of experiments, we apply the traditional and proposed version of KNN to the classification task, without decompose it into binary classifications, and use the accuracy as the evaluation measure. In this section, we observe the performance of the both versions of KNN, by changing the input size.

In Table I, we specify NewsPage.com, which is the text collection as the source for extracting classified words in this set of experiments. The text collection was used in the previous works for evaluating approaches to text categorization [8]. In each category, we extract 375 important words for building the collection of labeled words for evaluating the approaches to word categorization. In each category, the set of 375 classified words is partitioned into the 300 words as training examples and the 75 words as test examples, as shown in Table I. We select words by their frequencies concentrated in the given category combined with subjectivity in building the word collection.

Table I
THE NUMBER OF TEXTS AND WORDS IN NEWSPAGE.COM

Category	#Texts	#Training Words	#Test Words
Business	500	300	75
Health	500	300	75
Internet	500	300	75
Sports	500	300	75
Total	2000	1200	300

Let us mention the empirical process for validating the proposed approach to the task of word categorization. We extract the important words from each category in the above text collection, and encode them into numerical vectors. For each text example, the KNN compute its similarities with the 1200 training examples by the cosine similarity, and select the three most similar training examples as its nearest neighbors. Each of the 300 test examples is classified into one of the four categories: Business, Sports, Internet, and Health, by voting the labels of its nearest neighbors. The classification accuracy is computed by dividing the number of correctly classified test examples by total number of test examples, for evaluating the both versions of KNN.

Figure 8 illustrates the experimental results from categorizing the words using the both versions of KNN algorithm. The y-axis indicates the accuracy which is the rate of the correctly classified examples in the test set. Each group in the x-axis is the input size as the dimension of numerical and string vectors which represent texts. In each group, the gray and black bar indicate the performance of the traditional and proposed version of KNN algorithm, respectively. The most right group indicates the average over accuracies of the left four cases.

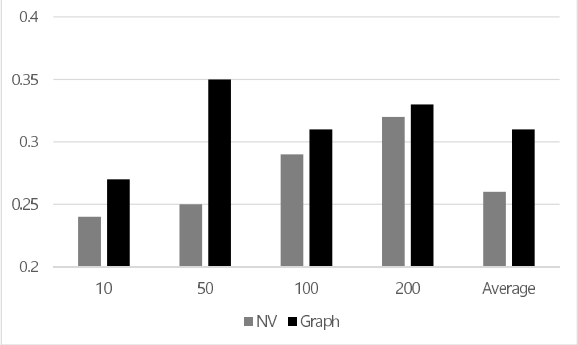

Figure 8. Results from Classifying Words in Text Collection: NewsPage.com

Let us make discussions on the results from doing the word categorization, using the both versions of KNN algorithm, as shown in Figure 8. The accuracy which are the performance measure of this classification task is in range between 0.24 and 0.44. The proposed version of KNN algorithm works better in the all input sizes; the accuracy of the proposed version reaches more than 0.4, in the input size 10. As the input size increases, the performance difference between both versions decreases; the performance of the traditional version improves proportional to the input size, but one of the proposed version stays around 0.35, except the input size 10. In this set of experiments, we conclude that the proposed version works outstandingly better than the traditional one, in averaging over the four cases.

B. Opinopsis

This section is concerned with the set of experiments for validating the better performance of the proposed version on the collection: Opinosis. In this set of experiments, the three categories are predefined in the collection, and we gather words category by category as the classified ones. Each word is classified exclusively into one of the three categories. The given classification is not decomposed into binary classifications and the accuracy is used as the evaluation measure. In this section, we observe the performances of the both versions of KNN algorithm with the different input sizes in the collection, Opniopsis.

In Table II, we illustrate the text collection, Opinosis, which is used as the source for extracting the classified words, in this set of experiments. The collection was used in previous works, for evaluating the approaches to text

categorization. We extract the 375 important words from each category as the collection of the classified words for evaluating the approaches to word categorization. In each category, as shown in Table 2, we partition the set of words into the 300 words as the training set and the 75 words as the test set. We select the words from the collection, depending on their frequencies which are concentrated on their own categories.

Table II
THE NUMBER OF TEXTS AND WORDS IN OPINIOPSIS

Category	#Texts	#Training Words	#Test Words
Car	23	300	75
Electronic	16	300	75
Hotel	12	300	75
Total	51	900	225

We perform this set of experiments by the process which is described in section IV-A. We extract the 300 important words by scanning individual texts in each category, and encode them into numerical vectors and string vectors, with the input sizes: 10, 50, 100 and 200. For each test example, the both versions of KNN computes its similarities with the 900 training examples and select the three most similar training examples as its nearest neighbors. Each of the 225 test examples is classified into one of the three categories, by voting the labels of its nearest neighbors. The classification accuracy is computed by the number of correctly classified test examples by the number of the test examples for evaluating the both versions of KNN algorithm.

In Figure 9, we illustrate the experimental results from categorizing the words using the both versions of KNN on this collection. Like Figure 8, the y-axis indicates the accuracy and the x-axis does the group of two versions by an input size. In each group, the grey bar and the black bar indicate the results of the traditional version and the proposed version of KNN algorithm, respectively. In Figure 2, the most right group indicates the average over results of the left three groups. Therefore, Figure 9 presents the results from classifying the words into one of the three categories by both versions of KNN algorithm, on the collection, Opinosis.

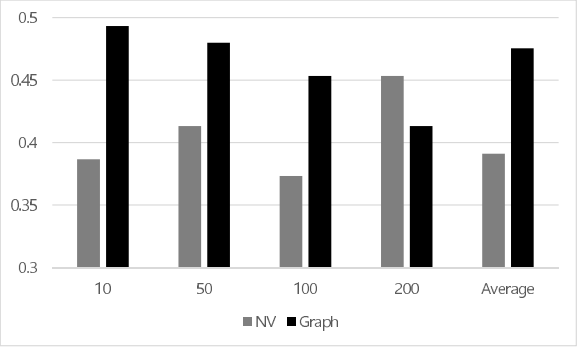

Figure 9. Results from Classifying Words in Text Collection: Opiniopsis

We discuss the results from doing the word categorization using the both versions of KNN algorithm, on Opinosis, shown in Figure 9. The accuracies of the both versions range between 0.35 and 0.75 in this task. The proposed version works better than the traditional one in the three input sizes: 10, 50, and 100. It is comparable with the traditional version in the other: 200. From this set of experiments, we conclude that the proposed one works outstandingly better in averaging over the four cases.

C. 20NewsGroups I: General Version

This section is concerned with one more set of experiments where the better performance of the proposed version is validated empirically on the text collection: 20NewsGroups I. In this set of experiments, we predefine the four general categories, and gather words from the collection category by category as the classified ones. Each word is classified exclusively into one of the four categories. We apply the KNN algorithms directly to the given task without decomposing it into binary classification, and use the accuracy as the evaluation measure. Therefore, in this section, we observe the performance of the both versions of KNN algorithm, with the different input sizes.

In Table III, we specify the general version of 20NewsGroups which is used for evaluating the two versions of KNN algorithm. In 20NewsGroup, the hierarchical classification system is defined with the two levels; in the first level, the six categories, alt, comp, rec, sci, talk, misc, and soc, are defined, and among them, the four categories are selected, as shown in Table III. In each category, we select 1000 texts at random, and extract 375 important words from them as the labeled words. The 375 words are partitioned into the 300 words as the training examples and the 75 words as the test ones, as shown in Table III. In the process of gathering the classified words, they are selected by their frequencies which are concentrated in their corresponding categories.

Table III
THE NUMBER OF TEXTS AND WORDS IN 20NEWSGROUPS I

Category	#Texts	#Training Words	#Test Words
Comp	1000	300	75
Rec	1000	300	75
Sci	1000	300	75
Talk	1000	300	75
Total	4000	1200	300

The experimental process is identical is that in the previous sets of experiments. In each category, we extract the 375 important words and encode them into numerical and string vectors with the input sizes, 10, 50, 100, and 200. For each test example, we compute its similarities with the 1200 training examples, and select the three similar ones as its nearest neighbors. The versions of KNN algorithm classify each of 300 test examples into one of the four categories: comp, rec, sci, and talk, by voting the labels of its nearest

neighbors. We also use the classification accuracy as the evaluation measure in this set of experiments.

In Figure 10, we illustrate the experimental results from categorizing words using the both versions on the broad version of 20NewsGroups. Figure 10 has the identical frame of presenting the results to those of Figure 1 and 2. In each group, the gray bar and the black bar indicates the achievements of the traditional version and the proposed version of KNN algorithm, respectively. The performance is expressed as the accuracy of classifying words into one of the four categories. In this set of experiments, the classification task is not decomposed into binary classifications.

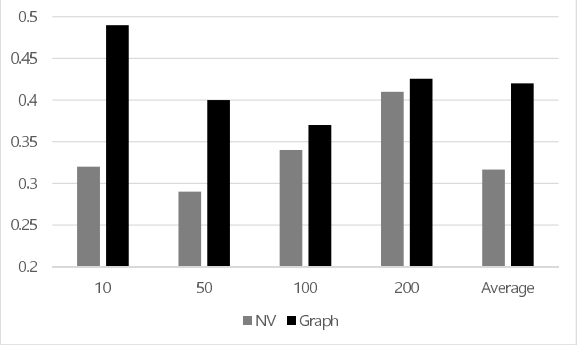

Figure 10. Results from Classifying Words in Text Collection: 20NewsGroup I

Let us discuss the results from doing the word categorization using the both versions on 20NewsGroups as shown in Figure 10. The accuracies of the both versions range between 0.28 and 0.47. The proposed version of KNN algorithm shows its better performances in the three of the four cases, but slightly less performance in the other. The inconsistent entries and the noisy values are the causes of degrading the performance of the proposed version, in the input size, 200. From this set of experiments, we conclude that the proposed version wins over the traditional one, in averaging over their four achievements, in spite of that.

V. Conclusion

Let us discuss the entire results from classifying words using the two versions of KNN algorithm. We compare the two versions with each other in the four collections. The proposed versions show its better results in all of the three collections. On the four collections, the accuracies of the traditional version range between 0.24 and 0.52, while, those of the proposed version range between 0.35 and 0.52. Finally, through the three sets of experiments, we conclude that the proposed version of KNN algorithm improves the word categorization performance, as the contribution of this research.

Let us mention some remaining tasks for doing the further research. We need to validate more the proposed approach in categorizing words in specific domains such as medicine, engineering, and economics, and customize it correspondingly. We need to consider other schemes of encoding words into graphs and other similarity measures between graphs. We modify other machine learning algorithms into their graph based versions where a graph is given by itself as the input data. We implement a word categorization system by adopting the proposed approach.

References

[1] K. Abainia, S. Ouamour, and H. Sayoud. “Neural Text Categorizer for topic identification of noisy Arabic Texts”, 1-8, Proceedings of 12th IEEE Conference on Computer Systems and Applications, 2015.

[2] D. Allemang and J. Hendler, Semantic Web for the Working Ontologies, Mrgan Kaufmann, 2011.

[3] D.J. Cook and L.B. Holder, Mining Graph Data, Wiley, 2007.

[4] T. Jo, “The Implementation of Dynamic Document Organization using Text Categorization and Text Clustering”, PhD Dissertation of University of Ottawa, 2006.

[5] T. Jo, “Table based Matching Algorithm for Soft Categorization of News Articles in Reuter 21578”, 875-882, Journal of Korea Multimedia Society, Vol 11, No 6, 2008.

[6] T. Jo, “Neural Text Categorizer for Exclusive Text Categorization”, 77-86, Journal of Information Processing Systems, Vol 4, No 2, 2008.

[7] T. Jo, “NTC (Neural Text Categorizer): Neural Network for Text Categorization”, 83-96, International Journal of Information Studies, Vol 2, No 2, 2010.

[8] T. Jo, “Normalized Table Matching Algorithm as Approach to Text Categorization”, 839-849, Soft Computing, Vol 19, No 4, 2015.

[9] T. Jo, “Semantic Word Categorization using Feature Similarity based K Nearest Neighbor”, 67-78, Journal of Multimedia Information Systems, 2018.

[10] T. Jo, “Encoding Words into Graphs for Clustering Word by AHC Algorithm”, 90-95, The Proceedings of 12th International Conference on Multimedia Information Technology and Applications, 2016.

[11] T. Jo, “Table based K Nearest Neighbor for Word Categorization in News Articles”, 1214-1217, The Proceedings of 25th International Conference on Computational Science & Computational Intelligence, 2018

[12] T. Jo, “Modification of K Nearest Neighbor into String Vector based Version for Classifying Words in Current Affairs”, 72-75, The Proceedings of International Conference on Information and Knowledge Engineering, 2018.

[13] T. Jo, “Extracting Keywords from News Articles using Feature Similarity based K Nearest Neighbor”, 68-71, The Proceedings of International Conference on Information and Knowledge Engineering, 2018.

[14] T. Jo, “Keyword Extraction in News Articles using Table based K Nearest Neighbors”, 1230-1233, The Proceedings of 25th International Conference on Computational Science & Computational Intelligence, 2018.

[15] T. Jo, “Modifying K Nearest Neighbor into String Vector based Version for Extracting Keywords from News Articles”, 43-46, The Proceedings of International Conference on Applied Cognitive Computing, 2018.

[16] T. Jo, “Index Optimization in News Articles using Feature Similarity based K Nearest Neighbor”, 106-109, The Proceedings of 17th Int’l Conference on e-Learning, e-Business, Enterprise Information Systems, and e-Government, 2018.

[17] T. Jo, “Optimizing Index of News Articles by Table based Version of K Nearest Neighbors”, 1238-1241, The Proceedings of 25th International Conference on Computational Science & Computational Intelligence, 2018.

[18] T. Jo, “String Vector based Version of K Nearest Neighbor for Index Optimization in Current Affairs”, 47-50, The Proceedings of International Conference on Applied Cognitive Computing, 2018.

[19] T. Jo, “Using Table based AHC Algorithm for clustering Words in Domain on Current Affairs”, 1222-1225, The Proceedings of 25th International Conference on Computational Science & Computational Intelligence, 2018.

[20] T. Jo, “Modification into Table based K Nearest Neighbor for News Article Classification”, 49-50, The Proceedings of 1st International Conference on Advanced Engineering and ICT-Convergence, 2018.

[21] T. Jo, “String Vector based AHC Algorithm for Word Clustering from News Articles”, 83-86, The Proceedings of International Conference on Information and Knowledge Engineering, 2018.

[22] T. Jo, “Improving K Nearest Neighbor into String Vector Version for Text Categorization”, 1091-1097, ICACT Transaction on Communication Technology, Vol 7, No 1, 2018.

[23] T. Jo, “Applying Table based AHC Algorithm to News Article Clustering”, 8-11, The Proceedings of International Conference on Green and Human Information Technology, Part I, 2019.

[24] T. Jo, “Introduction of String Vectors to AHC Algorithm for Clustering News Articles”, 150-153, The Proceedings of 21st International Conference on Artificial Intelligence, 2019.

[25] T. Jo, “Graph based Version of K Nearest Neighbor for classifying News Articles”, 4-7, The Proceedings of International Conference on Green and Human Information Technology Part I, 2019.

[26] T. Jo, “Graph based Version for Clustering Texts in Current Affair Domain”, 171-174, The Proceedings of 15st International Conference on Data Science, 2019.

[27] T. Jo and D. Cho, “Index Based Approach for Text Categorization”, 127-132, International Journal of Mathematics and Computers in Simulation, Vol 2, No 1, 2007.

[28] R. J. Kate and R. J. Mooney, ”Using String Kernels for Learning Semantic Parsers”, pp913-920, Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics, 2006.

[29] C. S. Leslie, E. Eskin, A. Cohen, J. Weston, and W. S. Noble, ”Mismatch String Kernels for Discriminative Protein Classification”, pp467-476, Bioinformatics, Vol 20, No 4, 2004.

[30] H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins, ”Text Classification with String Kernels”, pp419-444, Journal of Machine Learning Research, Vol 2, No 2, 2002.

[31] N.F. Noy and C. D. Hafner, “State of the Art in Ontology Design”, AI Magazine, Vol 18, No 3, 1997.

[32] L. Vega and A. Mendez-Vazquez, “Dynamic Neural Networks for Text Classification”, 6-11, The Proceedings of International Conference on Computational Intelligence and Applications, 2016.